VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Návrh a implementácia mechanizmu pre distribúciu MAC adries medzi GNU/Linux prepínačmi

Extension of the Routing Protocol for Distribution of the MAC Address Between GNU/Linux Software Switches

2011

Lukáš Margeťák

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

V Ostrave 25. júna 2011 .....

Rád by som na tomto mieste poďakoval všetkým, ktorí mi s prácou pomáhali, pretože bez nich by táto práca nevznikla. Predovšetkým mojej rodine, v ktorej som vždy mal plnú podporu, mojej láske, vedúcemu a môjmu dlhoročnému spolubývajúcemu.

Abstrakt

V práci je popísaná analýza ARP protokolu za účelom eliminácie jeho záplavového šírenia na linkovej vrstve, ďalej analýza smerovacích protokolov IS-IS, RIP, OSPF, BGP ako možnosť ich využitia pri tvorbe náhradného riešenia záplavového šírenia ARP, avšak na sieťovej vrstve.

Kľúčové slová: Smerovací protokol, ARP, Quagga, sieť, server, klient, prepínač, IS-IS, RIP, OSPF, BGP

Abstract

This thesis contains analyses of the ARP protocol with the aim of elimination of ARP flooding on the data link layer, then analyses of routing protocols such as IS-IS, RIP, OSPF, BGP as an option of using those to find a solution of this ARP flooding problem, moving the needed information transport to the network layer.

Keywords: Routing protocol, ARP, Quagga, network, server, client, switch, IS-IS, RIP, OSPF, BGP

Zoznam použitých skratiek a symbolov

ARP	–	Address Resolution Protocol
AS	–	Autonomous System
BGP	–	Border Gateway Protocol
CAM	–	Content Addressable Memory
CIDR	–	Classless Inter-Domain Routing
CSNP	–	Complete Sequence Number Packet
DIS	–	Designated Router
EGP	–	Exterior Gateway Protocol
EIGRP	–	Enhanced Interior Gateway Routing Protocol
ES	–	End System
IGP	–	Interior Gateway Protocol
IP	–	Internet Protocol
IPv4	–	Internet Protocol version 4
IPv6	–	Internet Protocol version 4
IS-IS	–	Intermediate System To Intermediate System
ISO/OSI	–	International Standardization Organization/Open Systems Interconnection
IPS	–	Internet Service Provider
LAN	–	Local Area Network
LSA	–	Link State Acknowledgment
LSP	–	Link State Packet
LSDB	–	Link State Database
MAC	–	Media Access Control
MED	–	Multi-Exit Discriminator
MTU	–	Maximum Transmission Unit
OSPF	–	Open Shortest Path First
PDU	–	Protocol Data Unit
PSNP	–	Partial Sequence Number Packet
RIP	–	Routing Information Protocol
RIPng	–	Routing Information Protocol next generation
TLV	–	Type Length Value
VLSM	–	Variable-Length Subnet Masking
VTY	–	Virtual Terminal

Obsah

Zoznam tabuliek

Zoznam obrázkov

Zoznam výpisov zdrojového kódu

1 Úvod

Cieľom tejto diplomovej práce je navrhnúť a implementovať rozšírenie vhodného smerovacieho protokolu o možnosť distribúcie naučených MAC adries medzi softwarovými prepínačmi realizovanými na platforme GNU/Linux. Vzniknuté riešenie by malo eliminovať záplavové šírenie ARP dotazov medzi softwarovými prepínačmi v rámci prepínaného segmentu. V úvode tejto práce je najprv analyzovaný protokol ARP a popísané možnosti správy lokálnych ARP tabuliek ako GNU/Linux prepínačov, tak aj staníc. Ďalej sa bližšie zoznámime s balíkom smerovacieho software Quagga, ktorý slúži ako simulátor smerovacích protokolov na platformách GNU/Linux. Je detailnejšie popísaný spôsob práce Quaggy, taktiež jednoduchá inštalácia a konfigurácia pre jej používanie. V kapitole 4 sa pozrieme na smerovacie protokoly, ktoré Quagga ponúka, ich analýzu a postrehy smerom k ich využitiu pre distribúciu naučených MAC adries medzi softwarovými prepínačmi. V kapitole 5 je uvedený návrh implementácie konečného riešenia ako aj popis samotnej implementácie. V záverečnej kapitole je zhrnutý výsledok tejto práce a jej prínos.

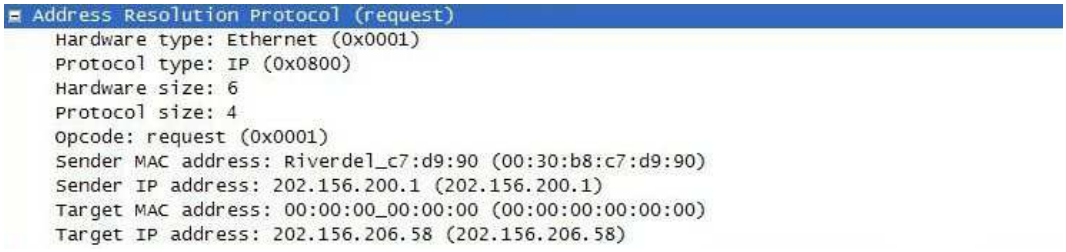
2 ARP

ARP je servisný protokol linkovej vrstvy. Poskytuje všeobecný mechanizmus, ktorý umožňuje preklad logických adries protokolov sieťovej vrstvy na fyzické MAC adresy spojovej vrstvy. Aj keď tento protokol bol pôvodne navrhnutý na preklad logických adries na fyzické adresy v sieti Ethernet, je použiteľný aj na vyhľadávanie fyzických adries v iných typoch sietí, ktoré podporujú všesmerové vysielanie (šírenie paketov ku všetkým uzlom v danej podsieti). Každý počítač si udržuje vlastnú ARP tabuľku s naučenými väzbami IP - MAC adries. Pri komunikácii pomocou protokolov vyšších vrstiev je známa logická adresa cieľa. Pre doručenie správy k cieľovému počítaču je však potrebné poznať jeho fyzickú adresu. Preklad adries sa vykonáva pomocou prekladovej tabuľky (ARP cache) s položkami tvaru <typ protokolu, protokolová adresa, prislúchajúca fyzická adresa >. Ak tabuľka takýto záznam neobsahuje, počítač rozošle všesmerovým vysielaním ARP požiadavku na MAC adresu cieľovej IP adresy. Keďže je to všesmerové vysielanie, dostane sa ku každej stanici v lokálnom segmente siete a každá stanica preskúma jeho obsah. Ak je požadovaná IP adresa zhodná s adresou stanice, stanica odpovie zaslaním ARP Reply. Počítač, ktorý vyslal všesmerové vysielanie si tak aktualizuje svoju ARP tabuľku a odošle správu cieľovému počítaču (adresovanému cieľovou MAC adresou, práve zistenou z ARP).

Ak sa záznam s fyzickou adresou cieľového uzlu v tabuľke nenachádza, informuje sa vyššia vrstva, že sa daný paket zahodí a vygeneruje sa nový paket nižšej vrstvy, ktorý bude obsahovať paket typu ARP.

V prípade mapovania 32 bitovej IP adresy na 48 bitovú MAC adresu sa vygeneruje nový paket.

- Hardware type: Typ fyzického rozhrania (napr. pre Ethernet je to 1)
- Protocol type: Typ protokolu (0x0800 pre IP)
- Hardware size: Veľkosť fyzickej adresy v B (pre Ethernet 6B)
- Protocol size: Veľkosť sieťovej adresy v B (pre IP 4B)
- Opcode: Typ operácie (požiadavka=1, odpoveď=2)
- Sender MAC address: Fyzická adresa odosielateľa
- Sender IP address: Sieťová adresa odosielateľa
- Target MAC address: Fyzická adresa cieľa (ak je známa - toto pole vyplňuje odosielateľ odpovede ”reply sender”)
- Target IP address: Sieťová adresa cieľa



Obr. 1: ARP požiadavka

2.1 Správa CAM/ARP tabuliek

Lokálne CAM/ARP tabuľky(spravované protokolom ARP na lokálnom počítači) je možné spravovať či už pridávaním statických IP - MAC väzieb, ich odoberaním alebo aktualizáciou. O všetko sa stará ARP prokokol sám, je však možné konfigurovať tieto tabuľky aj manuálne a urýchliť tak konvergenciu sieťového segmentu tým, že nebude nutné pre každú cieľovú IP adresu vysielať ARP Request, ale tieto väzby budú nakonfigurované manuálne(staticky).

3 Quagga

Quagga je balíček smerovacieho softwaru, ktorý poskytuje smerovacie služby využívajúc smerovacie protokoly RIPv1, RIPv2, RIPng, OSPFv2, OSPFv3, IS-IS ako s podporou IPv4, tak aj IPv6. Má interaktívne užívateľské rozhranie pre každý smerovací protokol a podporuje bežné klientské príkazy, používané pri konfigurácii smerovačov a prepínačov, ako aj GNU/Linux staníc. Vďaka tomuto dizajnu je možné ľahko Quaggu rozšíriť o nové protokoly.

Nainštalovaný a správne nakonfigurovaný systém Quagga sa správa ako dedikovaný smerovač. Vymieňa si teda smerovacie informácie s ostatnými smerovačmi cez smerovacie protokoly. Využíva tieto informácie k aktualizácii kernel smerovacej tabuľky, takže dáta smerujú vždy na správne miesto určenia. Je možné dynamicky meniť konfiguráciu a taktiež sledovať smerovaciu tabuľku z užívateľského rozhrania Quagga. Je nutné nastaviť IP adresy jednotlivým rozhraniam a potom v prípade malých sietí smerovať napríklad pomocou statických ciest, alebo naopak pri rozsiahlejších sieťach využiť výhody dynamických smerovacích protokolov ako RIP, BGP, OSPF.

Konfigurácia založená na UNIXových príkazoch sa vykonáva príkazmi route a ifconfig. Obsah smerovacej tabuľky je zobrazený pomocou príkazu nestat. Väčšina týchto príkazov funguje len ked má užívateľ správcovské privilégiá. Quagga však využíva úplne iný administratívny systém. Existujú dva užívateľkské módy:

- Normal mód: užívateľ môže sledovať len stav systému, nemôže zasahovať do konfigurácie
- Enable mód: užívateľovi je dovolené meniť systémovú konfiguráciu

```
zebra: Deklarácia rozhraní a statické smerovanie
bgpd: BGP smerovací protokol
ospfd: OSPF smerovací protokol
ospf6d: OSPF IPv6 smerovací protokol
ripd: RIP v2 smerovací protokol
ripngd: RIP Ipv6 smerovací protokol
isis: IS−IS smerovací protokol
```

Výpis 1: Možnosti nastavení Quaggy

3.1 Konfigurácia Quaggy

Vzhľadom k tomu, že mnou vytvorené rozšírenia Quaggy nie sú momentálne zahrnuté v distribúcii Quaggy na jej oficiálnych serveroch, je nutné Quaggu nainštalovať a nakonfigurovať z priloženého CD. Po skopírovaní zdrojových súborov je nutné v hlavnej zložke Quagga spustiť autokonfiguráciu simulátora pomocou príkazu configure. Po konfigurácii je potrebné skompilovať Quaggu pre Váš systém. To sa udeje pomocou príkazu make v hlavnej zložke Quagga. Po tomto kroku nasleduje samotná inštalácia, ktorá pozostáva z kopírovania skompilovaného programu a podporných súborov do štandardnej

lokality. Po tom, ako je inštalácia kompletná, súbory sú skopírované do /usr/local/bin a /usr/local/etc.

V rámci Quaggy je možné spúšťať jeden alebo viac smerovacích protokolov. Pred spustením Quaggy je však nutné tieto protokoly špecifikovať v súbore /etc/quagga/daemons a Quaggu znovu spustiť. Ako ukazuje Výpis 2, pri spustení Quaggy bude aktivovaná Zebra a smerovací protokol IS-IS. Viac o jednotlivých moduloch vo Výpise 1.

```
zebra=yes
bgpd=no
ospfd=no
ospf6d=no
ripd=no
ripngd=no
isis=yes
```

Výpis 2: Obsah súboru /etc/quagga/daemons

Ďalším krokom v konfigurácii Quaggy je vytvorenie konfiguračných súborov pre každý modul, v /etc/quagga/. Každý modul je asociovaný so špecifickým názvom súboru, ako ukazuje 3. Postačujúcou podmienkou k úspešnému behu Quaggy je aby tieto súbory boli prázdne. Nakoniec je potrebné priradiť správne práva pre súbory v adresári /etc/quagga:

```
#chown quagga.quaggavty /etc/quagga/*.conf
#chmod 640 /etc/quagga/*.conf
```

```
zebra: zebra.conf
bgpd: bgpd.conf
ospfd: ospfd.conf
ospf6d: ospf6d.conf
ripd: ripd.conf
ripngd: ripngd.conf
isis: isis.conf
```

Výpis 3: Asociácie modulov so súborovými názvami

Predvolené nastavenie Quaggy nedovoľuje pristupovať k jednotlivým modulom vzdialene. Na sprístupnenie vzdialeného prístupu (Telnet) je nutné upraviť konfiguračný súbor debian.conf. K adrese loopback rozhrania je nutné pripísať aj adresy z ktorých chceme vzdialene pristupovať.

```
hostname quagga-router
username root nopassword
```

Výpis 4: Obsah vtysh.conf

Každý bežiaci modul je možné spravovať pripojením sa cez telnet k správnemu portu, čo však nie je veľmi praktické(Výpis 5). Kôli zjednodušeniu správy bolo vytvorené takzvané VTYSH rozhranie. Jednotlivé moduly Quaggy majú svoje vlastné rozhrania alebo VTY linky pre komunikáciu s nimi. Po inštalácii je nutné nastaviť tieto porty, aby sa neskôr bolo možné pripojiť k jednotlivým modulom. Pre používanie VTY liniek, je nutné

najprv vytvoriť ich konfiguračný súbor /etc/quagga/vtysh.conf, ktorého obsah je zobrazený vo Výpise 4 a taktiež je potrebné nastaviť správne prístupové práva.

```
zebrasrv        2600/tcp        # zebra service
zebra           2601/tcp        # zebra vty
ripd            2602/tcp        # RIPd vty
ripngd          2603/tcp        # RIPngd vty
ospfd           2604/tcp        # OSPFd vty
bgpd            2605/tcp        # BGPd vty
ospf6d          2606/tcp        # OSPF6d vty
ospfapi         2607/tcp        # ospfapi
isisd           2608/tcp        # ISISd vty
```

Výpis 5: Nastavenie portov pre komunikáciu s modulmi

Po všetkých týchto nastaveniach môžeme bez problémov a s plnou funkcionalitou Quaggu prevádzkovať.

4 Smerovací protokol

Jedným z riešení záplavového šírenia ARP požiadaviek je práve využitie Quaggy a jej smerovacích protokolov na distribúciu či už ARP tabuliek samotných, alebo iných, vhodných konštrukcií. Pre tento spôsob riešenia je potrebné zvoliť najvhodnejší smerovací protokol, ktorý umožňuje prenášanie takýchto konštrukcií. Je vhodné, aby protokol umožňoval buď možnosť vytvoriť si vlastné konštrukcie, alebo nejakým spôsobom podporoval prenos MAC a IP adries, ktoré by po spracovaní mohli vytvoriť ARP záznam. V nasledujúcich podkapitolách sú popísané najznámejšie smerovacie protokoly z hľadiska možnosti ich využitia na distribúciu potrebných konštrukcií.

4.1 Všeobecne o smerovaní

Smerovanie je proces smerovania paketov zo zdrojového uzla do cieľového uzla v inej sieti. Posielanie paketov k ich next-hopu vyžaduje od smerovača vykonať dve základné operácie: zistenie a výber cesty a prepínanie paketov. Zistenie, prípadne výber cesty zahŕňa preskúmanie všetkých ciest do cieľovej siete a výber optimálnej trasy. Pre stanovenie optimálnej trasy sú informácie vložené do smerovacie tabuľky, ktorá zahŕňa informácie ako cieľovej siete, next-hop a súvisiace metriky. Prepínanie paketov zahŕňa zmenu cieľovej linkovej adresy na adresu next-hopu(sieťová cieľová a zdrojová adresa ostane nezmenená).

Informácie, ktoré smerovač potrebuje poznať, aby mohol smerovať paket:

- Cieľová adresa
- Susedné smerovače
- Možné cesty do vzdialených sietí
- Najlepšiu cestu ku každej sieti
- Ako udržiavať a verifikovať smerovacie informácie

Smerovače sa dozvedajú o vzdialených sieťach od susedných smerovačov alebo od administrátora. Na základe týchto informácií si potom smerovače vytvárajú smerovacie tabuľky, v ktorých sú uložené informácie o cestách do vzdialených sietí. Cesty sú buď priamo pripojené, statické alebo dynamické. Statické cesty sú nakonfigurované administrátorom. Dynamické cesty sa smerovač učí od susedných smerovačov pomocou smerovacích protokolov. Pri dynamickom smerovaní si smerovače aktualizujú informácie v navzájom dohodnutých intervaloch, prípade po zmenách v smerovacích tabuľkách. Ak smerovač dostane paket s cieľovou adresou, ktorá nie je v jeho smerovacej tabuľke, tento paket zahodí.

4.1.1 Statické versus Dynamické smerovanie

Statické smerovanie je proces pridávania ciest ručne, administrátorom. Ten konfiguruje cieľové siete, next-hopy a vhodné metriky. Cesta sa nezmení, pokým ju nezmení sám administrátor. Výhody statického smerovania:

- Žiadne nároky na procesor smerovača
- Žiadne využitie šírky pásma na linkách
- Bezpečnosť(len administrátor pridáva cesty)

Statické smerovanie má však aj svoje nevýhody:

- Ak je pridaná nová sieť, administrátor ju musí dokonfigurovať na všetkých smerovačoch
- Nemá praktické využitie v rozľahlých sieťach

Na rozdiel od statického smerovania, dynamické cesty sa prispôsobujú zmenám v prostredí siete automaticky. Keď dôjde k nejakým zmenám, smerovače začnú konvergovať prepočítavaním ciest a distribúciou aktualizácií. Správa s aktualizovanými informáciami sa šíri sieťou čím donúti ďalšie smerovače prepočítať svoje cesty. Proces končí keď skonvergujú všetky cesty. Tento proces využíva procesor smerovača a taktiež spotrebováva časť šírky pásma na linkách. Smerovacie protokoly definuje pravidlá pre smerovače o vzájomnej komunikácii.

Existujú dva typy smerovacích protokolov v sieťach, vnútorné smerovacie protokoly (IGP) a vonkajšie smerovacie protokoly (EGP). IGP sa používa v sieťach v spoločnom autonómnom systéme. EGP slúžia nau komunikáciu medzi AS. Ďalej sa smerovacie protokoly delia podľa typu algoritmu aký používajú na Distance-vector a Link-state a hybridné protokoly. Porovnanie dvoch najpoužívanejších typov je možné vidiet v tabuľke 1. Hybridné protokoly využívajú časti z Distance-vector aj z Link-state protokolov. Príkladom je EIGRP.

4.2 OSPF

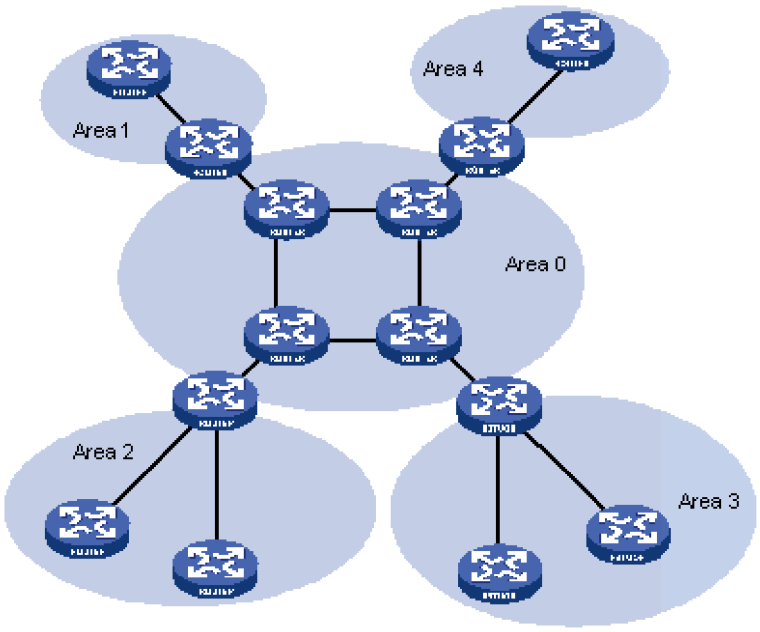
Open Shortest Path First (OSPF) je smerovací protokol pre IP siete. Využíva algoritmus smerovania na základe stavu liniek a spadá do skupiny vnútorných smerovacích protokolov, pôsobiacich v rámci jedného autonómneho systému.

OSPF je zrejme najpoužívanejší vnútorný smerovací protokol (IGP) v rozsiahlych podnikových sieťach. IS-IS, ďalší smerovací protokol založený na stave liniek, je častejšie využívaný vo veľkých sieťach poskytovateľov služieb. Najpoužívanejším vonkajším smerovacím protokolom je Border Gateway Protocol (BGP), hlavný smerovací protokol medzi autonómnymi systémami v sieti Internet.

OSPF je protokol, ktorý smeruje pakety len v rámci jednej smerovacej domény (autonómny systém). Zbiera informácie o stave liniek z dostupných smerovačov a buduje

Distance-vector	Link-state
Vidia sieť z perspektívy susedného smerovača	Vidia sieť z vlastnej perspektívy
Metrika sa akumuluje od smerovača k smerovaču	Počíta najkratšiu cestu k ostatným smerovačom
Aktualizácie sa opakujú periodicky	Aktualizácie sú vyvolávané
Pomalá konvergencia	Rýchla konvergencia
Vysielanie smerovacej tabuľky susedným smerovačom	Vysielanie informácií o stavu liniek všetkým smerovačom

Tabuľka 1: Distance-vector versus Link-state smerovacie protokoly

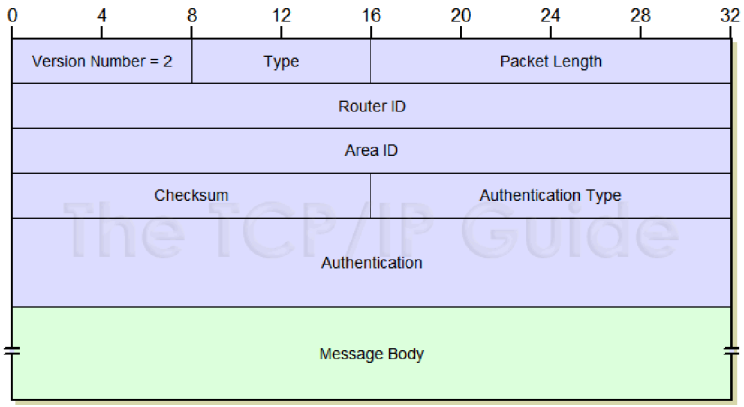


Obr. 2: Oblasti v OSPF

topologickú mapu siete. Topológia určuje smerovacie tabuľky, podľa ktorých robí smerovač smerovacie rozhodnutia, založené výhradne na cieľovej adresu IP nájdenej v IP paketoch. OSPF bol navrhnutý pre podporu Classless Inter-Domain Routing (CIDR) adresného modelu s premennou dĺžkou masky (VLSM). OSPF detekuje zmeny v topológii, ako je zlyhanie spojenia, veľmi rýchlo a konverguje na novú bez-slučkovú štruktúru v priebehu niekoľkých sekúnd. Počíta strom najkratších ciest pre každú cestu metódou založenou na Dijkstra algoritme. Informácie o stavoch liniek sú udržiavané na každom smerovači ako link-state databázy (LSDB), čo je stromový obraz celej topológie siete. Identické kópie LSDB sú pravidelne aktualizované prostredníctvom záplavového šírenia na všetky OSPF smerovače. OSPF smerovacou politikou na konštruovanie smerovacích tabuliek sa riadia faktory nákladov linky(externé metriky), ktoré sú spojené s každým smerovacím rozhraním. Nákladové faktory môžu byť vzdialenosť smerovača (round-trip time), sieťová priepustnosť linky alebo dostupnosť linky a jej spoľahlivosť, vyjadrené ako celé bezjednotkové čísla. To poskytuje možnosť dynamického vyvažovania záťaže medzi smerovacími cestami s rovnakou cenou. OSPF siete možno štruktúrovať a deliť do oblastí(Obrázok 2), čím sa zjednodušuje administratíva a optimalizuje sa prevádzka a využívanie dostupných zdrojov. Oblasti sú označené 32-bitovými číslami, vyjadrených buď len v desiatkovej, alebo často v osmičkovej sústave, známej z IPv4 adresného zápisu.

4.2.1 Správy OSPF

OSPF používa päť rôznych typov správ pri vymieňaní si ako informácií o stavoch liniek, tak aj všeobecných, medzi smerovačmi v rámci autonómneho systému alebo oblasti.

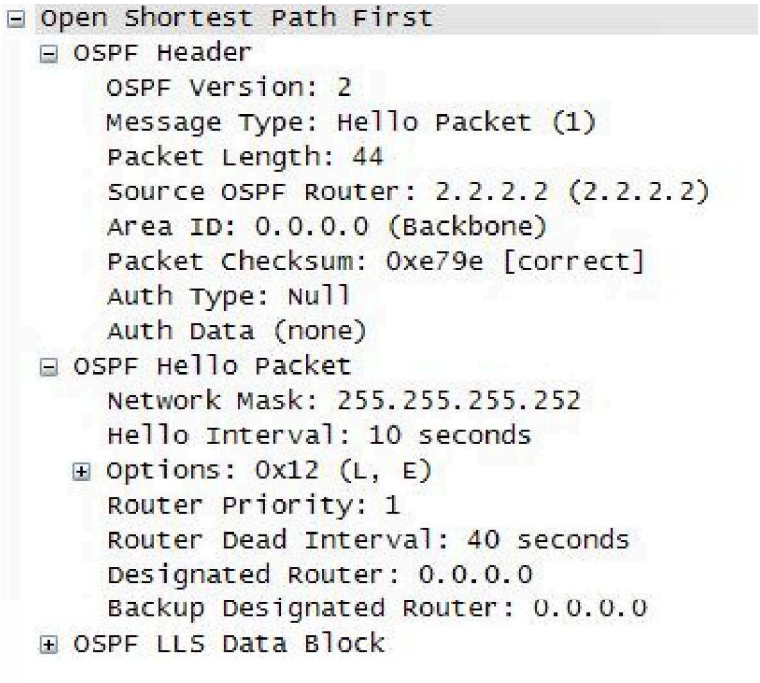


Obr. 3: Spoločný formát hlavičky OSPF

Každý typ OSPF správy obsahuje trochu iný súbor informácií. Všetky typy však zdieľajú podobnú štruktúru správ, počnúc spoločnou 24-bajtovou hlavičkou. Tá umožňuje transport určitých štandardne vyžadovaných informácií, ako je napríklad číslo verzie OSPF, ktorá správu vygenerovala. Umožňuje tiež prijímaciemu zariadeniu OSPF správy rýchlo zistiť typ obdržanej správy, takže vie, či je alebo nie je potrebné pokračovať skúmaním zvyšku správy(3).

Spoločná OSPF hlavička obsahuje:

- Version: Číslo verzie OSPF (2 pre verziu OSPF 2)
- Type: Typ OSPF správy
  1. Hello správa
  2. Database Description
  3. Link State Request
  4. Link State Update
  5. Link State Acknoledgment
- Packet Length: Dĺžka správy v bajtoch spolu s 24 bajtami samotnej hlavičky
- Router ID: ID smerovača, ktorý vygeneroval túto správu(IP adresa rozhrania, ktorým bola správa poslaná)
- Area ID: Identifikátor OSPF oblasti, do ktorej správa patrí
- Checksum: 16 bitový kontrolný súčet
- AuType: Indikuje typ autentikáciea použitej v správe
  1. Žiadna autentikácia (0)
  2. Jednoduchá autentikácia heslom (1)



Obr. 4: Formát OSPF Hello správy

3. Kryptografická autentikácia (2)

- Authentication: 64 bitové pole pre autentikáciu, ak je vyžadovaná

Za touto hlavičkou nasleduje telo správy, ktoré zahŕňa rôzny počet polí, v závislosti na type správy.

4.2.1.1 OSPF Hello správy:

Tieto správy majú hodnotu Type v hlavičke nastavenú na 1. Štruktúra správy je na Obrázku 4.

Hello správa obsahuje:

- Network mask: Maska podsiete, ktorej smerovač správu posiela
- Hello Interval: Počet sekúnd, ktoré smerovač čaká pred vyslaním ďalšej Hello správy
- Options: Ukazuje, ktoré voliteľné schopnosti OSPF smerovač podporuje
- Rtr Pri: Priorita smerovača, využívaná pri voľbe záložného designated smerovača
- Router Dead Interval: Počet sekúnd, počas ktorých musí byť smerovač bez odozvy, aby bol prehlásený za failed
- Designated Router: Adresa designated smerovača. Nastavené na 0 ak taký neexistuje

- Backup Designated Router: Adresa záložného designated smerovača
- Neighbors: Adresy všetkých susedov, od ktorých tento smerovač dostal Hello správu

4.2.1.2 OSPF Database Description správy: Tieto správy majú hodnotu Type v hlavičke nastavenú na 2. Database Description správa obsahuje:

- Interface MTU: Najväčšia možná veľkosť správy, ktorá môže byť vyslaná z tohto rozhrania smerovača bez fragmentácie
- Options: Ukazuje, ktoré voliteľné schopnosti OSPF smerovač podporuje
- Flags: Špeciálne vlajky hovoriace o vlastnostiach výmeny správ (I-initial, M-more after this one, MS-1 master, 0 slave)
- DD Sequence Number: Sekvenčné číslo Database Description správy
- LSA Headers: Obsahuje LSA hlavičky, ktoré nesú informácie o LSDB

4.2.1.3 OSPF Link State Request správy: Tieto správy majú hodnotu Type v hlavičke nastavenú na 3. Za hlavičkou nasleduje jeden alebo viac opakovaní nasledujúcich troch polí, každé opakovanie identifikujúce LSA, ktoré smerovač vyžaduje.

- LS Type: Typ požadovanej LSA
- Link State ID: Identifikátor LSA, väčšinou IP adresa smerovača alebo pripojenej siete
- Advertising Router: ID smerovača, ktorý vygeneroval LSA, ktorého aktualizácia je hľadaná

4.2.1.4 OSPF Link State Update správy: Tieto správy majú hodnotu Type v hlavičke nastavenú na 4. Tieto správy obsahujú len informáciu o počte LSA v tele správy nasledovanú jednotlivými LSA.

4.2.1.5 OSPF Link State Acknowledgment správy: Tieto správy majú hodnotu Type v hlavičke nastavenú na 5. Ďalej obsahujú zoznam LSA hlavičiek korešpondujúce s LSA, ktorých prijatie je tým potvrdené.

4.2.1.6 Link State Advertisements(LSAs): Ako ste si mohli všimnúť vyššie, niektoré typy správ obsahujú Link State Advertisemets, čo sú vlastne polia nesúce topologické informácie o LSDB. Je viac typov LSA, ktoré sa využívajú na transport informácií o rôznych typoch liniek(viac na obrázku 5). Každé LSA ma spoločnú 20 bajtovú hlavičku a za ňou ďalšie polia opisujúce linku. LSA hlavička obsahuje informácie postačujúce na identifikáciu linky, ako sú:

- LS Age: Počet sekúnd, ktoré uplynuli od vygenerovanie LSA
- Options: Ukazuje, ktoré voliteľné schopnosti OSPF smerovač podporuje
- LS Type: Typ linky, ktorú LSA popisuje
- Link State ID: Identifikátor linky, zvyčajne IP adresa smerovača alebo siete, ktorú linka reprezentuje
- Advertising Router: ID smerovača, pôvodcu LSA
- LS Sequence Number: Sekvenčné číslo
- LS Checksum: Kontrolný súčet LSA
- Length: Dĺžka LSA zahŕňajúca 20 bajtovú hlavičku

Za hlavičkou nasleduje telo LSA, obsahujúce špecifické polia na základe LS Type poľa. Z hľadiska významnosti pre výber vhodného smerovacieho protokolu na distribúciu potrebných dát tu uvádzam len sumárne informácie o týchto poliach:

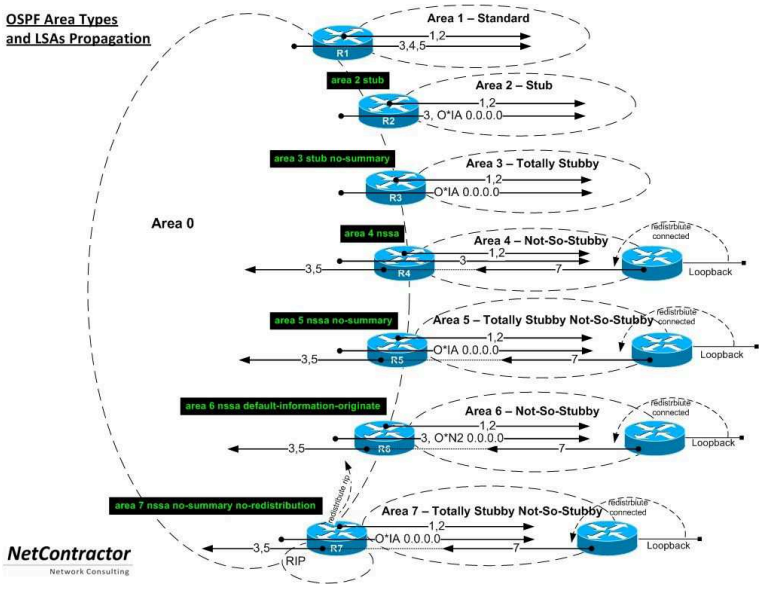
- Pre bežné linky k smerovaču, LSA obsahuje identifikátor smerovača a metriku, ktorou je možné ho dosiahnuť, ako aj detaily o smerovači(okrajový smerovač oblasti, hraničný smerovač)
- LSA pre siete obsahuje masku podsiete a informácie o ostatných smerovačoch v sieti
- Sumárne LSA obsahuje metriku a sumarizované adresy, ako aj masku podsiete
- Externé LSA obsahuje niekoľko ďalších polí dovoľujúcich externému smerovaču komunikovať

4.3 IS-IS

IS-IS je vnútorný smerovací protokol, navrhnutý pre používanie v rámci administratívnej domény alebo siete. IS-IS je smerovací protokol založený na stave linky. Posiela správy o stave liniek cez sieť smerovačov. Každý IS-IS smerovač si buduje obraz o sieťovej topológii nezávisle z obdržaných informácií. Podobne ako OSPF využíva Dijkstraov algoritmus na výpočet najlepších ciest v sieti. Pakety sú potom odoslané na základe spočítanej ideálnej cesty smerom k cieľovej stanici.

IS-IS smerovač sa nazýva Intermediate System (IS) a koncová stanice End System (ES). Protokol využívaný pre komunikáciu medzi ES a IS sa teda nazýva ES-IS protokol a protokol, pomocou ktorého môžu medzi sebou komunikovať smerovače sa nazýva IS-IS protokol.

V rámci OSI domény môže byť definované jedna alebo viac oblastí. Oblasť je logická entita, je formovaná skupinou susediacich smerovačov a liniek, ktoré ich spájajú. Všetky smerovače v oblasti si vymieňajú informácie o všetky koncových staniciach, na ktoré majú dosah. IS-IS smerovače sú navrhnuté tak, aby zapadali do niektorej z oblastí:



Obr. 5: Typy LSA

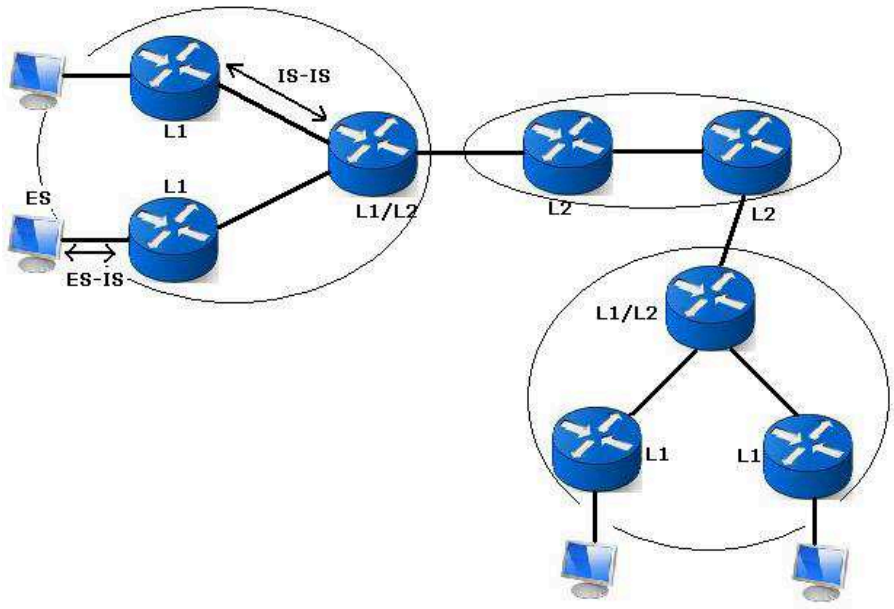
- Level 1 (intra-area): smerovače v rámci jednej oblasti, nemajú žiadne spojenie s inou oblasťou. Udržiavajú spoločnú link-state databázu.
- Level 2 (inter area): medzioblastný smerovať(medzi L1 oblasťami)
- Level 1-2 (both): slúži na prepojenie L1 a L2 smerovačov. Udržiavajú separované L1 a L2 link-state databázy.

Level 2 smerovače sú medzioblastné smerovače, ktoré môžu nadviazať susedskú väzbu s iným Level 2 smerovačom. Smerovacie informácie si vymieňajú len smerovače v jednej oblasti (Level 1 s Level 1, Level 2 s Level 2). Smerovače z Level 1-2 si vymieňajú smerovacie informácie s obidvoma levelmi (1 aj 2) a využívajú sa tak na prepojenie medzioblastných smerovačov a smerovačov v rámci jednej oblasti. Hranice medzi IS-IS oblasťami sú tak medzi smerovačmi Level 2 alebo Level 1-2. Výsledkom je, že smerovač je vždy súčasťou len jedinej oblasti. Skupina L2 smerovačov (vrátane L1/L2 smerovačov) a ich prepojenie tvorí IS-IS chrbticovú sieť(viď Obrázok 6). Každý L1 smerovač v rámci oblasti udržiava link-state databázu. L1/L2 smerovače nepropagujú L2 cesty pre L1 smerovače. Aby bolo možné smerovať paket do inej oblasti, L1 smerovač musí smerovať paket k L1/L2 smerovaču.

4.3.1 IS-IS Správy

V ISO terminológii sa pakety nazývajú Protocol Data Units (PDUs). Sú tri kategórie IS-IS správ:

- IS-IS Hello správy (IIHs): zakladajú a udržiavajú susedstvá medzi IS-IS susedmi.



Obr. 6: Typy IS-IS smerovačov

```
⊟ ISO 10589 ISIS InTRA Domain Routeing Information Exchange Protocol
    Intra Domain Routing Protocol Discriminator: ISIS (0x83)
    PDU Header Length  : 27
    Version (==1)      : 1
    System ID Length   : 0
    PDU Type           : L1 HELLO (R:000)
    Version2 (==1)     : 1
    Reserved (==0)     : 0
    Max.AREAs: (0==3)  : 0
```

Obr. 7: IS-IS hlavička

- Link State PDUs (LSPs): správy zodpovedné za distribúciu smerovacích informácií medzi IS-IS uzlami.
- Sequence Number PDUs (SNPs): tieto správy kontrolujú distribúciu LSP paketov. Poskytujú mechanizmus na synchronizáciu link-state databáz medzi smerovačmi v rámci spoločnej oblasti.

Každá kategória obsahuje rôzne typy správ. Každému typu je priradené Type number. Všetky IS-IS správy sú odosielané na multicastovú adresu v sieti LAN. Pre Level-1 smerovače sú pakety odosielané na 01-80-C2-00-00-14, pre Level-2 na 01-80-C2-00-00-15. Tabuľka 2 ukazuje rôzne typy IS-IS paketov.

IS-IS paket má hlavičku veľkosti 8 bajtov. Obsahuje nasledujúce položky:

Kategória	Typ paketu	Type number
Hello	LAN Level-1 Hello	15
	LAN Level-2 Hello	16
	Point-to-point Hello	17
LSP	Level-1 LSP	18
	Level-2 LSP	20
SNP	Level-1 Complete SNP	24
	Level-2 Complete SNP	25
	Level-1 Partial SNP	26
	Level-2 Partial SNP	27

Tabuľka 2: Typy IS-IS paketov

- Interdomain Routeing Protocol Discriminator: Identifikátor sieťovej vrstvy pridelený pre IS-IS. Jeho hodnota je 0x83 hexadecimálne.
- PDU Header Length: Špecifikuje veľkosť PDU hlavičky v bajtoch(bytes).
- Version: Nastavené na 1.
- System ID Length: Indikuje pole System ID of NSAP addresses.

  0 znamená dĺžku 6 bajtov.

  255 oznamuje dĺžku 0 bajtov.
- PDU Type: Obsahuje PRU Type Number poukazujúci na typ PDU.
- Version2: Tiež nastavené na 1.
- Maximum Area Addresses: Definuje maximálny počet synonymických oblastných adries, ktoré je možné využiť v L1 oblasti.

  0 znamená, že IS podporuje len tri oblastné adresy(by default).

  Akékoľvek číslo od 1 do 254 znamená počet povolených adries.

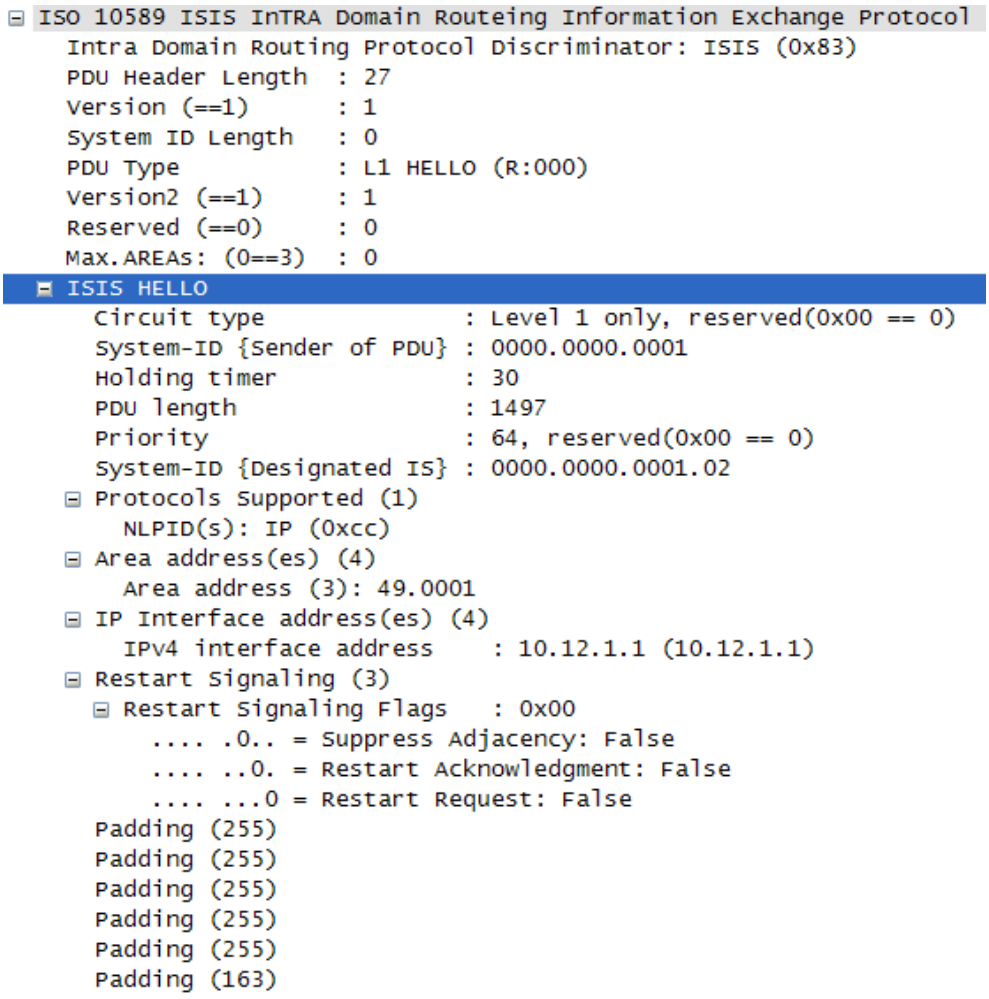
4.3.1.1 IS-IS Hello správy: IS-IS Hello správy sa využívajú na objavovanie susedov na pripojených linkách. Keď je sused objavený, správajú sa ako keepalive správy na udržiavanie susedstva. IS-IS štandard odporúča aby boli IS-IS Hello správy vtesnané do o jeden oktet menšej správy ako je veľkosť MTU. Sú dva typy Hello správ: LAN Hello a Point-to-point Hello správy. LAN Hello správy sa ďalej rozdeľujú na dva typy: Level-1 a Level-2 LAN Hello. Obidva typy LAN Hello správ majú rovnaký formát.

- Circuit Type: Typ smerovača.

  01: Level-1

  10: Level-2

  11: Level-1-2



Obr. 8: IS-IS Hello PDU

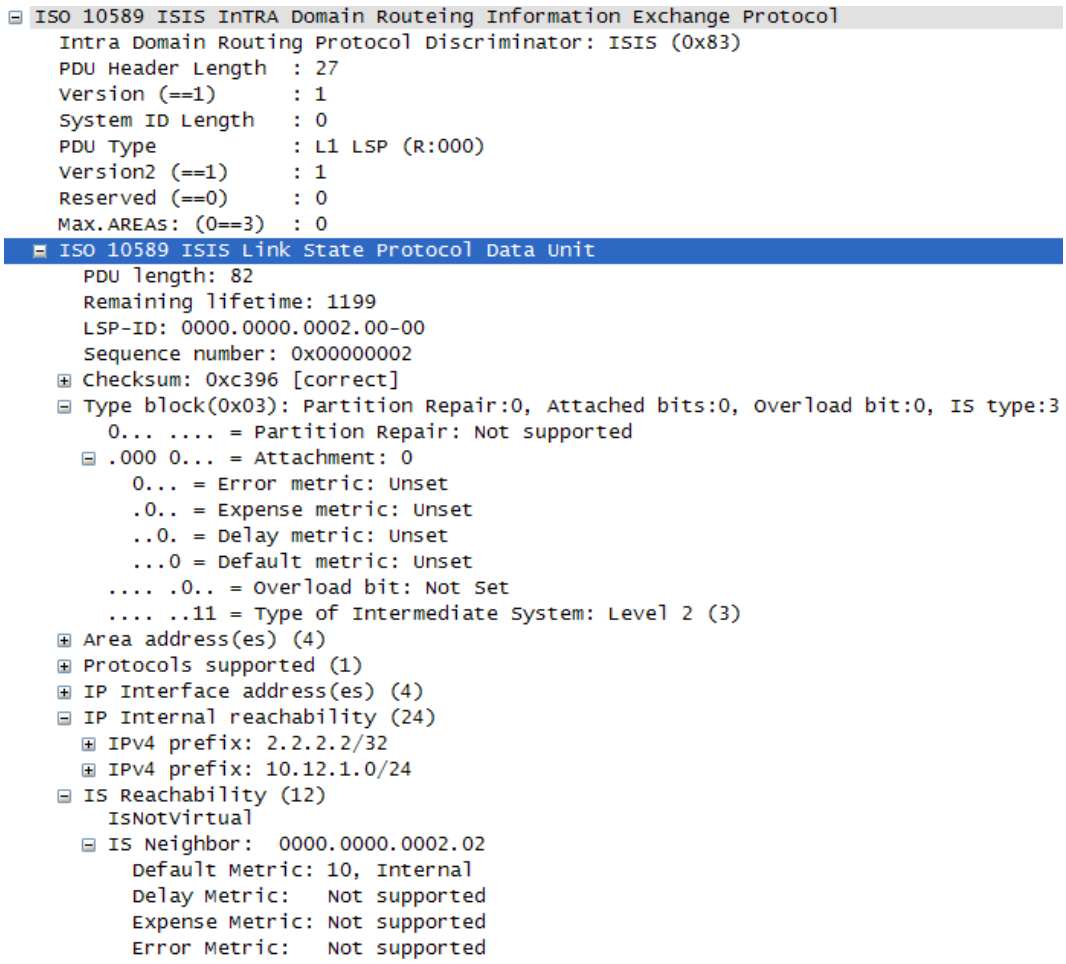
- Source ID: Pôvodca Hello správy (MAC adresa).
- Holding Timer: Perióda, ktorú má sused čakať na Hello správu skôr ako vyhlási suseda za mŕtveho.
- PDU Length: Veľkosť celého PDU v bajtoch(bytes)
- Priority: Nesie hodnotu medzi 0 a 127, ktorá je využítá pre voľbu DIS (Designated IS). Predvolená hodnota je 64.
- LAN ID: Hodnota tohto poľa je System ID plus Pseudonode ID.

Hello správy môžu prenášať ďalšie TLV, ako napríklad:

- Area Address(es): Obsahuje oblastné adresy nakonfigurované na smerovači, čo znamená, že na jednom smerovači môže byť nakonfigurovaných viac adries.
- IS Neighbor(s): TLV využívané len pre LAN Hello správy. Level-1 LAN Hello správy zaznamenávajú len L1 susedov, Level-2 LAN Hello zaznamenávajú len L2 susedov. IS Neighbor(s) prenáša MAC adresu susedov na lokálnej sieti.
- Protocols Supported: Prenáša Network Layer Protocol ID (NLPID) podporovaných protokolov. Pre IP protokol je to hodnota 0x81.
- IP Interface Address(es): Obsahuje IP adresu rozhrania z ktorého bolo PDU poslané.
- Padding: Využíva sa na doplnenie Hello PDU na aspoň minimálnu povolenú veľkosť. Cisco IOS nastavuje tieto bity na nulu.

4.3.1.2 IS-IS Link State správy: IS-IS využíva LSP na distribúciu a výmenu smerovacích informácií medzi IS-IS uzlami. IS-IS smerovač zaplavuje oblasť LSP správami kvôli identifikovaniu susedstiev, ich stavov a adresných oblastí, na ktoré má dosah. Formát L1 a L2 správ je rovnaký.

- PDU Length: Veľkosť celého PDU v bajtoch
- Remaining Lifetime: Čas v sekundách, počas ktorého je LSP považované za platné. Cisco IOS používa čas 20 minút (1200 sekúnd).
- LSP ID: System ID, Pseudonode ID a LSP číslo aktuálneho LSP.
- Sequence Number: Sekvenčné číslo LSP.
- Checksum: Kontrolný súčet obsahu LSP.
- Partition Repair (P-bit): Tento bit nie je podporovaný v Cisco IOS.



Obr. 9: IS-IS Link State PDU

- Attachment (ATT): 4-bitové pole. Cisco IOS však využíva len 1 bit. Slúži na oznámenie či je pôvodca správy(smerovač) pripojený k jednej alebo viacerým oblastiam.
- Overload (OL) bit: Ak má vysielajúci smerovač preplnenú pamäť, nastaví tento bit na 1. Prijímajúci smerovač potom nebude využívať tento smerovač pre tranzit.
- IS Type: Ukazuje, či smerovač, ktorý vyslal správu je L1 alebo L2.

  01- Level-1

  11- Level-2

L1 a L2 pakety obsajú aj ďalšie TLV okrem už spomínaných Area Address(es), Protocols Supported, IP Interface Address(es), ako napríklad:

- IP Internal reachability: Obsahuje priamo pripojené IP adresy spolu s maskami v rámci smerovacej domény. Taktiež obsahuje metriku.
- IS Reachability: Obsahuje IS-IS susedov (zahŕňajúc Pseudonodes) a metriku každej linky k týmto susedom.

L2 LSP paket taktiež obsahuje nasledujúce TLV:

- IP External reachability: Obsahuje IP adresy spolu s maskami, ktoré nie sú v smerovacej doméne a môžu byť dosiahnuté niektorým z rozhraní pôvodného smerovača.
- Inter-domain Routing Protocol Information: Toto TLV umožňuje L2 LSP transparentný prenos informácii z externých smerovacích protokolov cez IS-IS doménu.

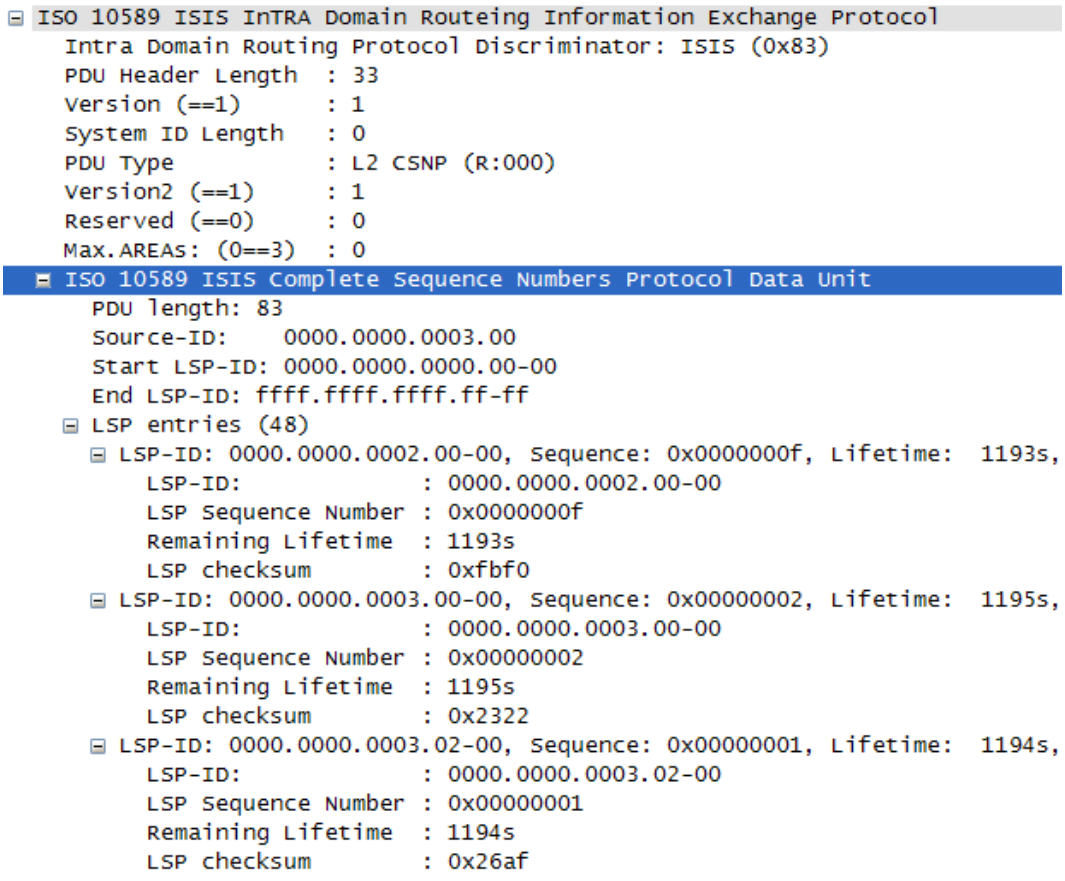
4.3.1.3 IS-IS Sequence Number správy: CSNP správy sa využívajú na udržiavanie LSDB. DIS periodicky posiela multicastové správy, v ktorých posiela všetky LSP. L1 SNP správy sa posielajú všetkým Level-1 IS na multicastovú adresu 01-80-C2-00-00-14, zatiaľ čo L2 SNP zase všetkým Level-2 IS s multicastovou adresou 01-80-C2-00-00-15.

Ak je databáza rozsiahla, nemôžu byť všetky LSP zahrnuté v jednej SNP správe. Vďaka tomu existuje Start LSP ID a End LSP ID pole. Ak jedna SNP správa obsahuje všetky informácie, Start LSP ID je nastavené na 0000.0000.0000.00-00 a End LSP ID na FFFF.FFFF.FFFF.FF-FF.

PSNP je veľmi podobné CSNP správam, avšak posiela len niektoré LSP, nie celú databázu. Na peer-to-peer sieti sa PSNP využíva na potvrdenie prijatej LSP správy. na LAN sieti je PSNP využívané k vyžiadaniu chýbajúcich alebo nových LSP správ.

Obidva druhy SNP správ obsahujú nasledujúce TLV pole:

- LSP Entries: TLV sumarizujúce LSP zoznamom, obsahujúcim Remaining Lifetime, LSP ID, Sequence Number a Checksum.



Obr. 10: IS-IS Sequence Number PDU

4.4 BGP

BGP - Border Gateway Protocol je protokol zodpovedný za smerovanie Internetu. Udržiava smerovacie tabuľky IP sietí a prefixov, ktoré určujú dostupnosť sietí medzi autonómnymi systémami. BGP je označovaný ako path-vector protokol pretože nevyužíva tradičnú metriku vnútorných smerovacích protokolov, ale robí smerovacie rozhodnutia na základe ciest, sieťových vlastností(policies) a/alebo pravidiel pre filtrovanie. Vzniklo ako náhrada za Exterier Gateway Protocol (EGP), protokol umožňujúci plne decentralizované smerovanie kvôli prechodu z hlavného ARPAnet modelu na decentralizovaný systém, ktorý zahŕňal NSFNET chrbticové siete a jeho združené regionálne siete. To umožnilo, aby sa stal internet skutočne decentralizovaným systémom. Od roku 1994 sa používa štvrtá verzia BGP. Všetky predchádzajúce verzie sú už zastarané. Medzi hlavné vylepšenia vo verzii 4 patrí podpora beztriedneho medzi doménového smerovania a využitie agregácie ciest, ktoré zmenšuje veľkosť smerovacích tabuliek. Väčšina poskytovateľov internetových služieb musí využívať BGP na prevádzku smerovania medzi sebou a ďalšími ISP. Preto, aj keď ho väčšina užívateľov internetu nepoužíva priamo, je BGP jedným z najdôležitejších protokolov Internetu.

4.4.1 Verzie BGP

BGP-1:

- Dĺžka správy v rozmedzí od 8 do 1024 bytov

BGP-2: Táto verzia už nepoužívala *up*, *down*, a *horizontal* vzťahy medzi autonómnymi systémami, ktoré boli použité vo verzii 1. BGP-2 uviedlo taktiež koncept atribútov ciest.

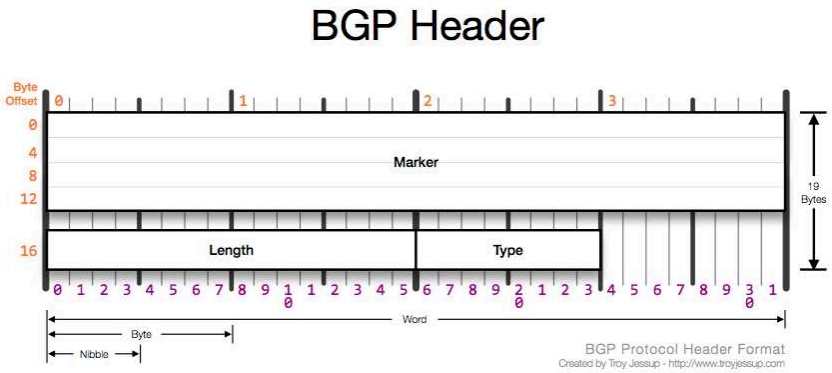
- Dĺžka správy v rozmedzí od 19 do 4096 bajtov

BGP-3: Verzia 3 protokolu BGP ruší niektoré obmedzenia pre používanie atribútu cesty NEXT_HOP, a pridáva pole identifikátor BGP do BGP OPEN správy(jednotlivé BGP správy v podkapitole 4.4.2). Taktiež objasňuje postup pre distribúciu BGP ciest medzi BGP susedmi v rámci autonómneho systému.

- Dĺžka správy v rozmedzí od 19 do 4096 bajtov

BGP-4: Táto verzia umožňuje reprezentovať viac classful sietí v jednej položke. Atribút AS_PATH bol upravený tak, aby mohli byť popísané skupiny autonómnych systémov, rovnako ako aj individuálne AS. Okrem toho bol obnovený INTER–AS METRIC atribút ako MULTI–EXIT diskriminátor. Boli pridané atribúty LOCAL–PREFERENCE a AGGREGATOR.

- Dĺžka správy v rozmedzí od 19 do 4096 bajtov
- Podporuje CIDR



Obr. 11: Hlavička BGP správy

4.4.2 BGP správy

BGP protokol posiela štyri typy správ:

- Open
- Update
- Keepalive
- Notification

Všetky BGP správy majú rovnakú, fixnú veľkosť hlavičky, ktorá obsahuje pole indikujúce celkovú dĺžku správy a typové pole označujúce typ správy(Obrázok 11).

4.4.2.1 BGP Open správy: Po tom, ako je založené spojenie medzi dvoma BGP systémami, tieto systémy si vymieňajú Open správy, aby tak vytvorili plné spojenie medzi nimi. Keď je spojenie založené, dva BGP systémy si môžu vymieňať BGP správy a dáta.

Open správa pozostáva z BGP hlavičky a týchto polí:

- Verzia: Aktuálna verzia BGP je 4
- Číslo lokálneho AS

- Navrhovaná hodnota Hold time časovača
- BGP identifikátor: IP adresa BGP systému
- Voliteľné parametre Dĺžka poľa a Parameter

4.4.2.2 BGP Update správy: BGP systém posiela aktualizačné správy na výmenu informácií o dostupnosti jednotlivých sietí. BGP potom využíva tieto informácie na skonštruovanie grafu, ktorý popisuje vzťahy medzi všetkými známimy AS.

Update správa pozostáva z BGP hlavičky a týchto polí:

- Unfeasible routes length: Dĺžka polí, ktoré obsahujú cesty vyradené z prevádzky, pretože už nie sú považované za dostupné
- Withdrawn routes: IP adresové prefixy pre tieto cesty
- Total path attribute length: Dĺžka polí, ktoré obsahujú atribúty ciest
- Path attributes: Atribúty ciest, zahŕňajúce pôvod cesty, atribút MED a informácie o agregácii, komunitách
- Network layer reachability information (NLRI): IP adresné prefixy použiteľných ciest propagovaných v aktualizačných správach

4.4.2.3 BGP Keepalive správy: BGP systémy si vymieňajú keepalive správy pre priebežné zisťovanie, či linka alebo host spadli alebo už nie sú dostupné. Tieto správy musia byť vymieňané dosť často na to, aby nevypršal timer Hold časovač. Pozostávajú len z BGP hlavičky.

4.4.2.4 BGP Notification správy: Tieto správy sú vysielané BGP systémom vtedy, keď je detekovaný chybový stav. Po tom, ako je správa poslaná, BGP a TCP spojenie medzi BGP systémami sú zatvorené. Tieto správy pozostávajú z BGP hlavičky spolu s chybovým kódom a subkódom, a dát, ktoré opisujú túto chybu.

4.4.3 BGP atribúty

S trasami, ktoré sú naučené prostredníctvom BGP sú spojené vlastnosti, ktoré sa používajú na určenie najlepšej cesty k cieľu v prípade, že existuje viac ciest. Tieto vlastnosti sa nazývajú BGP atribúty. Delia sa na známe(well–known) a voliteľné(optional). Ich rozdelenie je v tabuľkách 3 a 4.

K netranzitívnym BGP atribútom sa BGP systém správa tak, že ak ich pozná, tak ich pošle ďalej, ak ich však nepozná, zahodí ich. Analogicky je to s tranzitívnymi atribútmi, avšak v tomto prípade ich posiela ďalej všetky, bez rozdielu.

Atribút	Typ
Origin	Povinný
AS_Path	Povinný
Next–hop	Povinný
Local–pref	Voliteľný
Atomic–aggregate	Voliteľný

Tabuľka 3: Známe BGP atribúty

Atribút	Typ
MED	Netranzitívny
Originator ID	Netranzitívny
Cluster list	Netranzitívny
Aggregator	Tranzitívny
Community	Tranzitívny

Tabuľka 4: Voliteľné BGP atribúty

4.5 RIP

Routing Information Protocol (RIP) je distance-vector smerovací protokol, ktorý pracuje s počítaním preskokov ako so smerovacou metrikou. Jeden preskok sa rovná jednému prechodu smerovačom. RIP predchádza smerovacím slučkám implementáciou limitovaného počtu povolených preskokov na ceste od zdroja k cieľu. Maximálny počet preskokov pre RIP je 15. Limitovaný počet preskokov však tiež obmedzuje veľkosť sietí, ktoré môžu RIP podporovať. Počet 16 preskokov je považovaný za nekonečnú vzdialenosť a slúži na zrušenie neprístupných, nefunkčných, alebo inak nežiadúcich ciest počas výberového procesu. RIP implementuje split horizont, route poisoning a holddown mechanizmy, aby tak predchádzal propagovaniu nekorektných smerovacích informácií.

Pôvodne každý RIP smerovač prenášal plné aktualizácie každých 30 sekúnd. V rannom nasadení RIPu boli smerovacie tabuľky dosť malé na to, aby vyťaženie nebolo signifikantné. Ako však veľkosť sietí rástla, bolo jasné, že vzhľadom k tomuto časovému intervalu bude každých 30 sekúnd veľmi vysoký nárast zaťaženia liniek, aj keby smerovače tieto aktualizácie inicializovali v náhodných časoch. Myšlienka bola, že v dôsledku náhodnej inicializácie by sa táto záťaž rozprestrela v čase, v praxi sa to však nepotvrdilo. Sally Floyd a Van Jacobson v roku 1994 ukázali, že bez malej randomizácie časovača aktualizácií sa časovače po čase opäť zosynchronizujú. Vo väčšine súčasných sieťových prostredí nie je RIP preferovanou voľbou pre smerovanie, keďže jeho doba konvergencie a škálovateľnosť sú veľmi ťažko porovnateľné s EIGRP, OSPF alebo IS-IS (posledné dva sú link-state smerovacie protokoly), a (bez RMTI) limitovaný počet preskokov taktiež výrazne obmedzuje veľkosť siete. Avšak, jeho výhodou je, že je ľahké ho nakonfigurovať, pretože RIP nevyžaduje žiadne parametre na smerovači, na rozdiel od iných protokolov. RIP je implementovaný na vrchole UDP protokolu ako jeho transportný protokol. Je mu pridelené číslo portu 520.

RIPv1	RIPv2
Distance-vector	Distance-vector
Maximálny počet preskokov je 15	Maximálny počet preskokov je 15
Triedny	Beztriedny
Všesmerové vysielanie	Multicastové vysielanie na adresu 224.0.0.9
Nepodporuje VLSM	Podporuje VLSM
Nepodporuje nespojité siete	Podporuje nespojité siete

Tabuľka 5: RIPv1 versus RIPv2

4.5.1 Verzie RIP

Všetky nižšie spomenuté verzie smerovacieho protokolu RIP sa stále používajú, väčšinou však pre výučbové účely alebo len v nie veľmi rozsiahlych a komplikovaných sieťach. Priame porovnanie verzií RIPv1 a RIPv2 je možné vidieť v tabuľke 5.

RIP version 1 (RIPv1): Jednoduchý distance-vector protokol. Je vybavený rôznymi technológiami ako napríklad Split Horizon a Poison Reverse, aby mohol podávať lepší výkon v komplikovaných sieťach.

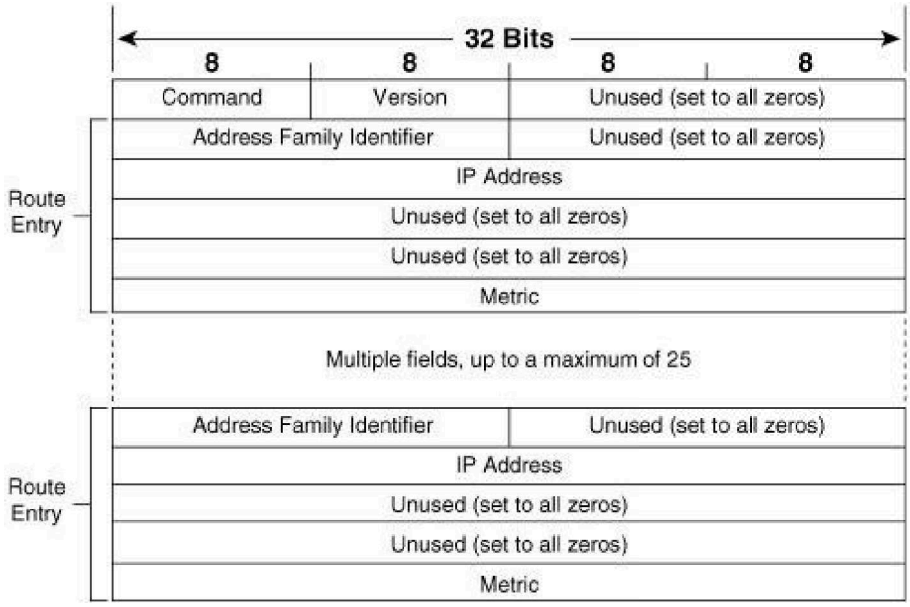
- Najdlhšia cesta nemôže prekročiť 15 preskokov
- RIP využíva statickú metriku pri porovnávaní ciest
- Maximálna dĺžka datagramu je 512 bajtov, nezahŕňajúc IP alebo UDP hlavičky.

RIP version 2 (RIPv2): Druhá verzia pridáva niekoľko nových čŕt:

- Tagovanie externých ciest
- Masky podsietí
- Adresy next-hop smerovača
- Autentikácia
- Podpora multicastov

RIPng: RIPng(next generation) je nadstavba RIPv2 protokolu pre podporu IPv6, protokolu novej generácie. Hlavné rozdiely medzi RIPv2 a RIPng sú:

- Podpora IPv6
- RIPv2 podporuje RIPv1 updaty, RIPng nie. V tom čase sa predpokladalo, že IPv6 smerovače budú využívať IPsec na autentizáciu
- RIPv2 umožňuje pripájanie ľubovoľných tagov k cestám, RIPng nie
- RIPv2 kóduje next-hop do každého smerovacieho záznamu, RIPng vyžaduje špecifické kódovanie next-hopov pre skupinu záznamov



Obr. 12: Hlavička RIP správy

4.5.2 RIP správy

Každá RIP správa obsahuje príkaz a číslo verzie a môže obsahovať položky až do 25 záznamov(ciest). Každá cesta obsahuje identifikátor adresnej rodiny, IP adresu dostupnú touto cestou a počet preskokov na trase. Ak musí smerovač odoslať viac ako 25 záznamov, musí byť vyprodukovaných viac RIP správ. Na obrázku 12 si môžeme všimnúť, že prvá časť správy je tvorená štyrmi oktetmi, a každá cesta je 20 oktetov. Preto je maximálna veľkosť správy 4 + (25 x 20) = 504 bajtov. Vrátane osem bajtovej hlavičky UDP bude maximálna veľkosť datagramu RIP (bez IP hlavičky) 512 bajtov.

Hlavička správy RIP protokolu obsahuje tieto položky:

- Command: Vždy nastavený buď na jednotku, čo znamená Request, alebo dvojku, čo znamenať správu s odpoveďou. Existujú aj ďalšie príkazy, ale všetky sú buď zastarané, alebo vyhradené pre súkromné použitie
- Version: Verzia bude nastavená na jednotku pre RIPv1, na dvojku pre RIPv2
- Address Family Identifier: Identifikátor adresnej rodiny je nastavená na dvojku pre IP. Jedinou výnimkou je požiadavka na plnú smerovaciu tabuľku smerovača (alebo hostiteľa)
- IP address: Adresa cieľa. Táto položka môže byť sieťová adresa, adresa podsiete
- Metric: Počet preskokov medzi 1 a 16

4.5.2.1 RIP Request správy: Request správy sú správy vyslané smerovačom inému smerovaču za účelom získania celej, alebo časti jeho smerovacej tabuľky.

4.5.2.2 RIP Response správy: Správy zaslané smerovačom obsahujúce všetky alebo časť svojej smerovacej tabuľky. Tieto správy nie sú odoslielané iba ako reakcia na správu RIP Request.

4.6 Resume

Quagga ako balík smerovacieho software ponúka široké možnosti čo sa týka využitia smerovacích protokolov, ako som spomenul v predchádzajúcom texte. Najrôznejšie smerovacie protokoly ponúkajú rozmanité možnosti. Z hľadiska vopred pripravených štruktúr sa ako najvýhodnejšie javí použitie smerovacieho protokolu IS-IS, ktorý, ako bolo spomenuté, ponúka možnosť využiť takzvané TLV polia, kde je možnosť definovať typ, dĺžku a hodnotu poľa. Je teda možné si upraviť tieto TLV polia podľa vlastnej potreby, v mojom prípade teda na distribúciu IP a MAC adries medzi IS-IS prepínačmi na platforme GNU/Linux. V ostatných smerovacích protokoloch by tiež bola možnosť posielať ľubovoľné dáta, avšak nemajú pripravené žiadne konštrukcie, čo znamená, že by sa všetko muselo doprogramovať a zakomponovať do bežiacich procedúr a funkcií smerovacieho protokolu.

5 Implementácia

Po analýze vlastností použiteľných smerovacích protokolov v kapitole 4 sa natíska otázka ako náročná, či už časovo alebo zložitosťou, je analýza tisícov riadkov zdrojových kódov za účelom ich ďalšej úpravy v porovnaní s návrhom a vlastnou implementáciu smerovacieho protokolu, ktorá sa bude starať len o distribúciu správ nutných k správnej funkcionalite systému, bez ďalšej zbytočnej záťaže. Ako je v tejto kapitole spomenuté, najvhodnejším kandidátom na distribúciu väzieb IP-MAC bez nutnosti veľmi obšírnej implementácie je smerovací protokol IS-IS. Má možnosť definovať si vlastné dátové polia pomocou TLV, kde je možné špecifikovať typ, veľkosť a obsah. Po snahe analyzovať správanie sa modulu IS-IS som však zistil, že bude menej časovo náročné implementovať vlastný smerovací protokol, ako upravovať zdrojové kódy niekoho iného. Samotný modul IS-IS obsahuje tisíce riadkov kódu a nie je v ňom jednoduchá orientácia aj vďaka tomu, že k nemu neexistuje žiadna programátorská príručka a komentárov v samotných zdrojových súboroch nie je postačujúce množstvo. Preto som sa v neskoršej fáze rozhodol implementovať vlastný proprietárny protokol, ktorý sa bude starať o výmenu potrebných informácií medzi jednotlivými prepínačmi.

Keďže je Linux napísaný v programovacom jazyku C, rozhodol som sa môj protokol programovať v rovnakom jazyku. Uľahčilo to tak prácu so systémovými súbormi a knižnicami. Pred samotnou implementáciou som analyzoval možnosti komunikácie unixových počítačov na sieti medzi sebou. Dostal som sa k často využívaným socketom, ktoré veľmi dobre zapadali do mojej koncepcie komunikácie na báze klient - server, kde server počúva na vopred stanovenom porte prípadný ohlas klienta, a po zachytení jeho signálu mu posiela svoju ARP tabuľku. Klient zase, vysiela na adresu a port serveru žiadosť o ARP tabuľku, ktorá mu je následne poslaná. Po tejto úvodnej komunikácii si úlohy vymenia a majú tak vymené ARP tabuľky. Vždy keď pribudne nový GNU/Linuxový prepínač, vymení si informácie s najbližšie pripojeným prepínačom, ktorý už komunikuje mojím protokolom. Idea bola taká, že si prepínače budú vymieňať tieto informácie vždy po nejakom časovom intervale, aby sieť nebola zbytočne zahlcovaná prípadnými výmenami po každej zmene.

5.1 Distribúcia ARP tabuliek

Aby sa ARP tabuľka mohla distribuovať, je nutné najprv lokálnu ARP tabuľku spracovať, vybrať z nej potrebné informácie a tie uložiť do vhodnej štruktúry. V prípade žiadosti o odoslanie ARP tabuľky je potrebné pripravenú štruktúru upraviť do formátu prenositeľného sieťou, po prijatí opäť tabuľku spracovať a uložiť potrebné dáta do lokálnej ARP tabuľky počítača. Vzhľadom k neskoršiemu rozhodnutiu o spôsobe prijímania a vysielania tabuliek (viac v podkapitole 5.3) som si vytvoril dve separované tabuľky so štruktúrou zobrazenou vo výpise zdrojového kódu 6. Tabuľka s písmenom I na konci slúži na uloženie prichádzajúcich dát z okolitých ARP tabuliek, a druhá tabuľka s označením O sa používa pre naplnenie dátami z lokálnej ARP tabuľky a ich následné odoslanie klientovi.

struct ARP_entryI

```
{
  char IPaddr[16];
  char MACaddr[18];
};

struct ARP_entryO
{
  char IPaddr[16];
  char MACaddr[18];
};
static struct ARP_entry ARP_tableI[ARP_TABLE_SIZE];
static struct ARP_entry ARP_tableO[ARP_TABLE_SIZE];
```

Výpis 6: Štruktúra ARP tabuľky

O naplnenie ARP tabuľky v prepínači sa stará funkcia, ktorá k získaniu potrebných dát využíva súbor /proc/net/arp v linuxovom súborovom systéme, v ktorom je aktuálna ARP tabuľka uložená. Po štarte operačného systému tento súbor obsahuje len IP-MAC väzby priamo pripojených sieťových rozhraní. Tie si táto procedúra uloží do svojich vlastných štruktúr, aby s nimi mohla potom ďalej pracovať. V prípade žiadosti klienta o ARP tabuľku je táto jednoduchou funkciou spracovaná do textového reťazca. Funkciu je možné vidieť vo výpise zdrojového kódu 7. Vzhľadom k tomu, že v lokálnej ARP tabuľke nie sú uložené informácie o lokálnych rozhraniach, je vhodné tieto informácie do novo vzniknutej ARP tabuľky môjho protokolu dodať, preto som dopísal funkciu na získanie lokálnych IP a MAC adries.

```
void add_data(char *data, struct ARP_entryO *tableO, int line)
{
  int i=0;

  while (i < line)
  {
    strcat(data, tableO[i].IPaddr);
    strcat(data, " ");
    strcat(data, tableO[i].MACaddr);
    strcat(data, " ");
    i++;
  }
}
```

Výpis 7: Príprava ARP tabuľky na odoslanie

5.2 Spracovanie ARP tabuľky prepínačom

Po prijatí paketu s ARP tabuľkou je tento paket analyzovaný a ďalej spracovaný. Prijatý paket je rozparsovaný po jednotlivých adresách(IP a MAC), kde každá adresa je uložená do odpovedajúcej bunky, ako môžeme vidieť vo výpise zdrojového kódu 8. Funkcia vracia počet riadkov ARP tabuľky, aby nebolo nutné toto číslo znova zisťovať pri zapisovaní informácií do lokálnej ARP tabuľky.

```
int parse_buff(char *buf, struct ARP_entryI *tableI)
{
  char *obsah;
  int i = 0;

  obsah = strtok(buf," ");
  while (obsah != NULL)
  {
    strcpy(tableI[i].IPaddr, obsah);
    obsah = strtok(NULL, " ");
    strcpy(tableI[i].MACaddr, obsah);
    obsah = strtok(NULL, " ");
    i++;
  }
  return i;
}
```

Výpis 8: Spracovanie dát do ARP tabuľky

Aby s týmito informáciami poslanými od susedných staníc mohol pracovať samotný prepínač a využívať ich pri komunikácii či už so stanicami alebo ďalšími prepínačmi, je potrebné zapísať IP a MAC adresy do lokálnej ARP tabuľky. Je viac možností ako tento problém vyriešiť, vzhľadom k tomu že je môj protokol napísaný v jazyku C, bolo by vhodné v ňom napísať aj funkciu zápisu do lokálnej ARP tabuľky. Pri hľadaní vhodného riešenia som však narazil na problém. Existuje niekoľko nástrojov pracujúcich s lokálnou ARP tabuľkou, avšak v prípade, že by som chcel ich funkcionalitu využiť, musel by som k mojím zdrojovým súborom zahrnúť ďalších niekoľko desiatok súborov, z ktorých využijem len pár riadkov. Vzhľadom k možným zbytočným komplikáciám som sa rozhodol využiť to, čo je v každom unixovom systéme dostupné: shell. Vytvoril som si teda jednoduchý skript, ktorému predám parametre IP a MAC adresu, a on ich zapíše to lokálnej ARP tabuľky.

GNU platforma ako taká dokáže obsluhovať rôzne vstupno-výstupné operácie na veľa rôznych zariadeniach a objektov s využitím jednoduchých súborov - read(čítanie), write(zápis), lseek(posune čítacej alebo zapisovacej hlavy). Avšak, väčšina zariadení má pár špeciálnych operácií, ktoré do tohto modelu nezapadajú, ako napríklad:

- Zmena fontu na termináli
- Prevíjanie magnetickej pásky vpred alebo dozadu (nemôže sa pohybovať v inkrementoch bajtov, lseek je nepoužíteľný)
- Udržiavanie smerovacej tabuľky pre sieť(aj ARP tabuľky)

Aj keď niektoré objekty ako napríklad sockety a terminály majú ich vlastné špeciálne funkcie, nebolo by praktické vytvárať ich pre všetky tieto prípady. Namiesto toho komunikácia s kernelom prebieha cez takzvané ioctl() funkcie definované v sys/ioctl.h. Ioctl() funkcia má minimálne dva parametre, prvým je file-deskriptor (viac v podkapitole 5.3.1) a druhým je vstupno-výstupný príkaz, ktorý je na ňom vykonaný. Na výpise

zdrojového kódu 9 môžeme vidieť nutné prvotné vytvorenie socketu s parametrami adresná rodina socketu(IP), typ socketu(Stream alebo Datagram viď podkapitola 5.3.1) a protokol (0 pre lokálnu komunikáciu). Ioctl() funkcia je potom zavolaná práve na tento socket s príkazom SIOCSARP(pridať alebo modifikovať ARP záznam), ktorého parametrom je štruktúra req, kde je uložená IP a MAC adresa záznamu a takisto vlajka záznamu (0x02 pre kompletný záznam, 0x04 pre permanentný záznam).

```
struct arpreq req;

...

s = socket(AF_INET, SOCK_DGRAM, 0);
ioctl (s, SIOCSARP, (caddr_t)&req);
```

Výpis 9: Inicializácia socketu a volanie funkcie ioctl()

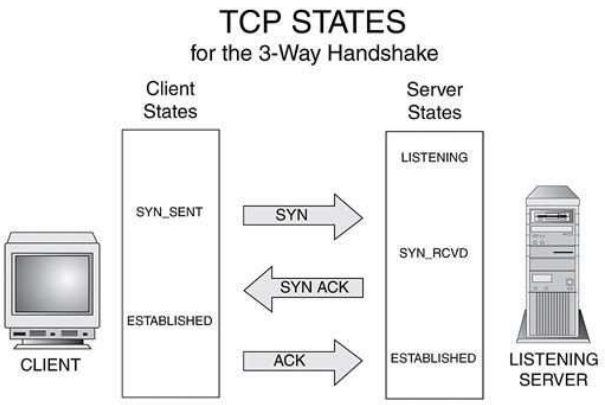
5.3 Odosielanie a prijímanie

Ako som už spomenul v úvode tejto kapitole, Unixové systému ponúkajú možnosť využiť takzvané sockety na komunikáciu cez sieťové rozhrania. Socket je prostriedok komunikácie medzi procesmi Unixových systémov. Od iných komunikačných prostriedkov sa líši hlavne tým, že tieto procesy nemusia bežať na jednom počítači. Ako takmer všetko v Unixovom systéme, aj socket je súbor. Procesy, ktoré chcú komunikovať medzi sebou využívajú špeciálne funkcie na prístup k tomuto súboru, a tak si jednoducho vymieňajú dáta v obidvoch smeroch. Jednoducho povedané, Unixový socket je bitový tok(stream) medzi procesmi, bežiaci či už lokálne, alebo na v rámci sieťou spojených Unixových systémoch. Z hľadiska mojich potrieb pri zaistení komunikácie na sieti je vhodné spomenúť dva typy socketov, aj keď je ich samozrejme viac. Tie dva spomínané typy sú teda Stream sockety a Datagram sockety.

5.3.1 Stream a Datagram Sockety

Stream sockety sú spoľahlivé, spojovo orientované komunikačné toky umožňujúce nie len komunikáciu medzi procesmi na rôznych staniciach, ale aj medziprocesnú komunikáciu na jednej stanici. Socket sa vytvorí zavolaním funkcie socket(), ktorá vracia filedescriptor rovnako ako napríklad funkcia open(). Na čítanie alebo zapisovanie do socketu môžeme využiť napríklad funkcie read() a write(). Navyše je možné použiť niekoľko funkcií špeciálne pre sockety. Preto som sa rozhodol ich využil aj pri pridávaní ARP záznamov, ako je spomenuté v podkapitole 5.2. Teraz sa však budem venovať socketom slúžiacim na komunikáciu po sieti. Ak pošlete na socket tri položky v poradí 0 1 2, tak na opačnom konci prídu v rovnakom poradí a taktiež bez chyby(error free). To, že dáta prídu bez chyby zaisťuje použitie protokolu TCP ktorý zaisťuje, že dáta sú doručené bez chyby a sekvenčne. Z aplikácií alebo protokolov, ktoré používajú Stream sockety by som spomenul dobre známy telnet a HTTP protokol ako príklady.

Datagramové sockety, niekedy nazývané aj nespojové sa tak nazývajú preto, že neexistuje žiadna garancia, že odoslaný datagram bude doručený, ani kedy bude doručený.



Obr. 13: Nadviazanie TCP spojenia

Isté však je, že ak bude doručený, tak bude taktiež bez chýb. Na rozdiel od Stream socketov, Datagram sockety využívajú UDP protokol. Nespojové sú preto, že nie je nutné spojenie medzi dvoma stanicami, ako je tomu pri Stream socketoch. Tu len pripravíte dáta, priložíte IP hlavičku s cieľovou adresou a dáta sú odoslané.

Pri implementácii môjho protokolu je priam nevyhnutné, aby boli dáta doručené v správnom poradí, bez chýb, a aby sa nestalo, že sa časť cestou stratí. Preto som si vybral Stream sockety, ktoré sú spojovo orientované a zamedzujú tak strate alebo veľkému oneskoreniu prijatia dát(pokiaľ nie je preťažená sieť).

5.4 Nadviazanie komunikácie

Každý GNU/Linux prepínač využívajúci môj protokol si udržiava zoznam susedov obsahujúci IP adresy jednotlivých komunikujúcich staníc. Je to jednoduchá štruktúra obsahujúca len IP adresy. Aby bolo možné spojenie nadviazať, musí byť k dispozícii práve adresa cieľa, ku ktorému sa chceme pripojiť. Je teda potrebné vyriešiť problém nadviazania komunikácie. Ako teda vyslať ARP tabuľku, keď neviem kam? Na to som si vytvoril funkciu, ktorá po spustení protokolu rozošle všesmerovým vysielaním lokálne IP adresy spolu s MAC, aby ich tak mohli zachytiť ostatné prepínače, a poslať na ne ARP tabuľku. Server teda obdrží jednu alebo viac IP adries, ktoré porovná so svojou databázou susedov a zistí tak, či už tieto adresu v zozname má, alebo nie. Ak tam niektoré adresy nemá, tak ich do zoznamu pridá a nadviaže s nimi TCP spojenie ako klient(Obrázok 13). Taktiež si prichádzajúce IP a MAC adresy uloží do lokálnej ARP tabuľky, aby pri zahájení komunikácie nemusel využívať ARP všesmerové vysielanie. Nadviazaným TCP spojením si klient vyžiada ARP tabuľku, server mu ju pošle a klient ju spracuje. Server, ktorý ARP tabuľku vyslal, hneď po tomto akte vyšle svoju lokálnu adresu, ktorú spracuje prepínač, ktorý práve prijal ARP tabuľku. Opäť ju porovná so svojou databázou a na základe odpovede na otázku či adresu v zozname má alebo nie sa rozhodne čo ďalej. Tento spôsob som aplikoval preto, že keď sa prepínač pripojí do siete neskôr, nemusí zachytiť všesmerové

vysielanie iného prepínača a tak nedostane potrebné dáta. Týmto spôsobom dá o sebe vedieť hneď po prihlásení, na čo môžu reagovať ostatné prepínače.

5.5 Paralelizácia

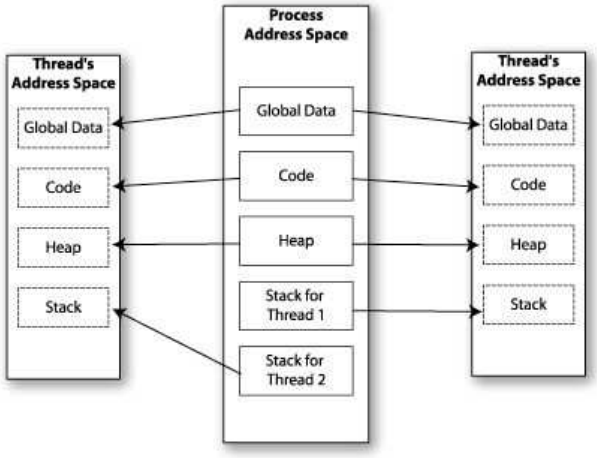
Aby mohol program využívajúci sockety súčasne počúvať na jednom porte, a vysielať na inom, je nutné beh programu paralelizovať. Na samotné sockety je možné využiť funkciu select() alebo poll(). Tieto funkcie sú schopné pracovať s tromi skupinami socketov:

1. skupina: sleduje, či sa v niektorom sockete nenachádzajú prichádzajúce dáta
2. skupina: sleduje, či nie je možné do niektorého socketu zapisovať dáta
3. skupina: sleduje, či sa niektoré sockety nenachádzajú v chybovom stave

Potreboval som však paralelizovať aj ďalšie operácie a výpočty, preto som sa nechcel obmedzovať len na sockety, ale na paralelizáciu ako takú. Ako zaujímavá možnosť s javilo využitie vlákien, kde by procesy bežali v samostatných vláknach. Technicky je vlákno definované ako nezávislý tok inštrukcií, ktorého beh si môže operačný systém naplánovať. Každé vlákno si udržiava vlastný ukazovateľ na zásobník, vlastné registre, vlastnosti plánovanie(priorita), vláknu špecifické dáta.

V Unixovom prostredí teda vlákno existuje v rámci procesu a využíva jeho zdroje, má však nezávislé riadenie toku. Duplikuje len najzákladnejšie zdroje potrebné k jeho nezávislému behu, môže však niektoré zdroje procesu zdieľať s inými vláknami, avšak sa stále správať rovnako nezávisle. Keďže vlákna v rámci procesu zdieľajú a využívajú jeho zdroje, zmeny spôsobené jedným vláknom sú viditeľné aj ostatným vláknam, taktiež dva ukazovatele majúce rovnakú hodnotu ukazujú na rovnaké dáta a v neposlednom rade čítanie a zapisovanie na rovnakú lokalitu v pamäti je možné, takže si to vyžaduje explicitnú synchronizáciu programátora. Aj to je teda dôvod, prečo vytvoriť pre každé vlákno vlastné dátové štruktúry(nie všetky).

Ako je vidieť vo výpise zdrojového kódu 10, hodnota funkcie pthread_create() vracia celočíselnú hodnotu, 0 ak je úspešná, v opačnom prípade vracia číslo, ktoré znamená chybu. Pri vytváraní vlákien je nutné špecifikovať funkciu, ktorú vlákno spúšťa a prípadné parametre. Príkaz pthread_join() čaká na skončenie obidvoch vlákien. Keby nebol použitý, program skončí po tom, ako spadne prvé vlákno, čo vlastne zabezpečuje beh programu do nekonečna a možnosť paralelných výpočtov. V prvom vlákne teda volám funkciu lineOne, ktorá zabezpečuje rozoslanie lokálnych adries všesmerovým vysielaním a potom počúva na porte asociovanom so všesmerovým vysielaním prichádzajúce adresy susedných prepínačov. V prípade, že nejaké adresy obdrží a nemá ich už v zozname susedov, spustí ARP klienta, ktorý sa pripojí k ARP serveru s cieľovou adresou a portom 3490, ktorú práve obdržal a prijme od neho ARP tabuľku. Druhé vlákno sa teda stará o to, aby bola žiadosť o ARP tabuľku obdržaná a vypočutá. Počúva teda na porte 3490, v prípade že dostane žiadosť, posiela späť žiadanú ARP tabuľku. Tretie vlákno spúšťa funkciu time, ktorá sa stará o aktualizácie, spomenuté v nasledujúcej časti.



Obr. 14: Proces versus Vlákno

```
pthread_t thread1, thread2, thread3;
int   iret1 , iret2 , iret3 ;

iret1  = pthread_create( &thread1, NULL, lineOne, (void *)IPlist ); //(void *) &
iret2  = pthread_create( &thread2, NULL, lineTwo, NULL);
iret3  = pthread_create( &thread3, NULL, time, (void *) IPlist );

pthread_join( thread1, NULL);
pthread_join( thread2, NULL);
pthread_join( thread3, NULL);
```

Výpis 10: Inicializácia vlákien

5.6 Aktualizácie

Možností ako aktualizovať ARP tabuľky medzi susednými prepínačmi je hneď niekoľko. Inšpiráciu ponúkajú aj smerovacie protokoly, kde sa využíva ako aktualizácia po vopred stanovenom časovom úseku, tak aktualizácie vyvolané zmenou smerovacej informácie. Z hľadiska zaťaženosti siete sa javí ako výhodnejšia možnosť aktualizovať po zmene, ak je však tých zmien príliš veľa, nie je to až taká veľká úspora. Na druhej strane aktualizácia po dohodnutom časovom intervale môže priniesť výhody v tom, že sieť je zaťažená len raz a to práve počas výmeny ARP tabuliek. Aby som mohol obísť nevýhodu vysokej záťaže na sieti v jeden moment, využil som možnosť počítať tento časový úsek na každom prepínači zvlášť. Interval sa teda odvíja od času spustenia protokolu pre výmenu ARP tabuliek, nie od času kedy dva prepínače nadviazali medzi sebou susedský vzťah a vymenili si prvé informácie, čo má za následok rozprestretie záťaže do širšieho časového spektra(v závislosti na časoch spustenia inštancií protokolu). Po ubehnutí časového inter-

valu teda TCP klient nadviaže spojenie so serverom a prijme aktualizovanú ARP tabuľku. Server opäť počúva na vopred dohodnutom porte, tentokrát 3495, a v prípade nadviazania spojenia si pripraví ARP tabuľku tak, že prečíta celú lokálnu ARP tabuľku a tú potom posiela klientovi.

5.7 Obsluha protokolu

Protokol v adresári src na priloženom CD je pripravený k používaniu na GNU/Linux platformách, či už staniciach alebo prepínačoch. Jediným krokom, ktorý treba pred samotným spustením protokolu spraviť je jeho kompilácia, čo sa učiní príkazom make priamo v unixovom shelli. Zdrojové súbory sú tak skompilované a pripravené k spusteniu. Počas kompilácie sú vytvorené dva spustiteľné súbory, súbor s názvom protocol, ktorým sa program spúšťa, a súbor s názvom stop, ktorým beh protokolu zastavíte. Kvôli práci so systémovými knižnicami je nutné protokol spúšťať s právami super-užívateľa. Po spustení súboru protocol je program spustený a beží na pozadí, takže nie je možné sledovať žiadne výpisy na konzole a beh protokolu tak užívateľovi nezasahuje do práce na konzole. Zastavenie behu protokolu je možné vykonať zadaním príkazu stop na konzolu a proces je hneď ukončený.

6 Záver

Zámerom tejto diplomovej práce bolo navrhnúť a implementovať rozšírenie vhodného smerovacieho protokolu o možnosť distribúcie naučených MAC adries medzi softwarovými prepínačmi realizovanými na platforme GNU/Linux a zamedziť tak záplavovému šíreniu ARP dotazov medzi softwarovými prepínačmi v rámci prepínaného segmentu v takej miere, že riešenie bude disponovať mechanizmom na distribúciu potrebných ARP informácií, avšak nie na linkovej vrstve, ale na vrstve sieťovej. Jednou z možností bolo využiť balík smerovacieho software Quagga, ktorý disponuje všeobecne známymi smerovacími protokolmi ako je RIP, IS-IS, OSPF, a využiť tieto protokoly na prenos potrebných informácií. Využitie týchto protokolov znamenalo pochopenie spôsobu ich práce na úrovni zdrojového kódu, jeho analýza a následný návrh konceptu dátových štruktúr mechanizmu v podobe rozšírenia aktuálnej verzie zdrojového kódu. Ako je spomenuté v podkapitole 4.6, niektoré smerovacie protokoly už disponujú voľne upraviteľnými dátovými štruktúrami v zmysle možnej špecifikácie dátovej štruktúry na prenos ľubovoľného formátu dát(IS-IS TLV), no v ostatných smerovacích protokoloch by bolo nutné implementovať všetko potrebné pre transport ARP dát od základu. Po dôkladnej analýze som sa teda rozhodol implementovať vlastný protokol, ktorý sa bude starať len o prenos ARP informácií na segmentoch siete. Protokol je dotiahnutý do takej fázy, že je schopný sám inicializovať a odpovedať na spojenia inštancií protokolu na ďalších prepínačoch v sieti, je schopný ARP informácie odoslať a spracovať do lokálnych ARP tabuliek a taktiež, ako je spomenuté v 5.6, využíva aj časované aktualizácie. Vzhľadom k možnostiam tohto protokolu využívať ho okrem GNU/Linux prepínačov aj na staniciach je protokol vyčlenený z balíka smerovacieho software Quagga a je spustiteľný a použiteľný bez nutnosti inštalácie a konfigurácie Quaggy, čo však považujem za prínos z hľadiska širšej využiteľnosti tohto protokolu nie len v softwarových prepínačoch, ale aj v staniciach realizovaných na platforme GNU/Linux.

7 Literatúra

[1] SCHRODER, Carla, *Linux Networking Cookbook,* USA : O Reilly Media, Inc., 2007. 640 s. ISBN 978-0-596-10248-7

[2] MOY, John T., *OSPF: anatomy of an Internet routing protocol.*, USA : Addison-Wesley, 1998. 368 s. ISBN 978-0-201-63472-3.

[3] GREDLER, Hannes; GORALSKI, Walter, *The complete IS-IS routing protocol*, Germany : Springer, 2005. 540 s. ISBN 978-1-85233-822-0.

A Obsah CD

Diplomová práca obsahuje vložené CD, na ktorom sa nachádza tento text v elektronickej podobe vo formátoch PDF a LaTeX, zdrojové kódy Quaggy a zdrojový kód s protokolom.

Adresár	**Podadresár**	**Obsah**
doc	dip	Text práce, formát PDF a LaTeX
src	Quagga	Balík smerovacieho software Quagga
	Protokol	Zdrojové súbory Protokolu

Tabuľka 6: Obsah CD